DOM-based XSS

In DOM-based XSS the malicious code is never sent to the server. The injection-point is somewhere where javascript has access.

The typical example of how this works is with URLs.

The user is able to control the URL with the help of the hash-symbol `#`. If we add that symbol to a URL the browser will not include that characters that comes after it in the requet to the server.

```
https://example.com/#this_is_not_sent_to_server
```

However, the complete URL is included in DOM-objects.

```
document.URL
# will generate this output: https://example.com/#this_is_not_sent_to_server
```

Source

So in order to inject and execute a DOM-based XSS we need a injection-point (called source) and a point of execution (called sink).

In the example above `document.URL` is our source. Example of other sources are:

```
    document.URL
    document.documentURI
    document.URLUnencoded (IE 5.5 or later Only)
    document.baseURI
    location
    location.href
    location.search
    location.hash
    location.pathname

    window.name
    document.referrer
```

Sinks

```
eval
setTimeout
setInterval
setImmediate
execScript
crypto.generateCRMFRequest
ScriptElement.src
ScriptElement.text
ScriptElement.textContent
ScriptElement.innerText
anyTag.onEventName
```

Finding it

To find DOM-based XSS you will need to check out the code.

If the javascript code is bundled and minified you can use js_beautify to make it readble again.

```
sudo apt-get install libjavascript-beautifier-perl
# then invoke js_beautify
```

References

https://github.com/wisec/domxsswiki/wiki/location,-documentURI-and-URL-sources